Romain Jacquet **BTS SIO 1**
25/11/2025

Compte Rendu TP POO (Programmation Orienté Objet)

Sommaire

Introduction

Dans ce TP, l'objectif était de créer des classes en Java pour gérer des livres, des clients et des comptes bancaires. Cela m'a permis d'apprendre à utiliser les bases de la programmation orientée objet, comme les classes, les méthodes.

Exercice 1

Etape 1

Création de la classe

J'ai fait une classe `Livre` avec trois attributs :

- `titre` (String)
- `auteur` (String)
- `prix` (int)

J'ai mis ces attributs en privé pour que personne ne puisse y toucher directement.

Etape 2

Dans ma classe, j'ai prévu plusieurs éléments :
J'ai créé un constructeur par défaut (`public Livre()`) pour pouvoir créer un objet *Livre* sans fournir de paramètres.

J'ai ajouté trois paires de getters et setters pour chaque attribut, afin de pouvoir lire et modifier leurs valeurs.

J'ai aussi créé une méthode `afficher()` qui permet d'afficher les informations du livre de façon claire.

Etape 3

Pour chaque attribut, j'ai mis en place des méthodes classiques :

Getters (permettent de récupérer la valeur) :

- `getTitre()` → j'obtiens le titre
- `getAuteur()` → j'obtiens l'auteur
- `getPrix()` → j'obtiens le prix

Setters (permettent de modifier la valeur) :

- `setTitre(String titre)` → je modifie le titre
- `setAuteur(String auteur)` → je modifie l'auteur
- `setPrix(int prix)` → je modifie le prix

Etape 4

Avec la méthode `afficher()`, j'ai la possibilité d'afficher toutes les informations du livre dans un format propre et lisible, par exemple :

```
Livre : [titre]
Écrit par : [auteur]
Prix : [prix] €
```

Etape 5

J'ai une classe qui contient une méthode `main()` pour démontrer son fonctionnement. Voici les étapes du test :

J'ai créé une instance de la classe `Livre` en utilisant `new Livre()`.
J'ai initialisé les attributs de l'objet à l'aide des méthodes `setters` appropriées.
J'ai appelé la méthode `afficher()` pour visualiser le résultat.

```
Livre : Le Petit Prince
�crit par : Antoine de Saint-Exup�ry
Prix : 12 €
```

Exercice 2

1.

J'ai créé la classe `Client` pour stocker les informations du client : CIN, nom, prénom, et téléphone.

```
class Client {
    private String cin;
    private String nom;
    private String prenom;
    private String tel;
}
```

2.

J’ai ajouté les getters et setters pour chaque attribut de la classe `Client`

```
public String getCin() { return cin; }
public void setCin(String cin) { this.cin = cin; }

public String getNom() { return nom; }
public void setNom(String nom) { this.nom = nom; }

public String getPrenom() { return prenom; }
public void setPrenom(String prenom) { this.prenom = prenom; }

public String getTel() { return tel; }
public void setTel(String tel) { this.tel = tel; }
```

3.

J’ai créé un constructeur prenant tous les attributs en paramètres.

```
public Client(String cin, String nom, String prenom, String tel) {
    this.cin = cin;
    this.nom = nom;
    this.prenom = prenom;
    this.tel = tel;
}
```

4.

J’ai ajouté un second constructeur pour initialiser uniquement `cin`, `nom` et `prénom`

```
public Client(String cin, String nom, String prenom) {
    this.cin = cin;
    this.nom = nom;
    this.prenom = prenom;
}
```

5.

J’ai implémenté une méthode `Afficher()` dans la classe `Client`.

```
public void Afficher() {
    System.out.println("CIN: " + this.cin);
    System.out.println("Nom: " + this.nom);
    System.out.println("Prénom: " + this.prenom);
    System.out.println("Tél: " + this.tel);
}
```

6.

J’ai créé la classe `Compte` avec un attribut `static dernierCode` pour générer un code unique à chaque compte.

```
public class Compte {
    private static int dernierCode = 0;
    private int code;
    private int solde;
    private Client proprietaire;
}
```

7.

J’ai ajouté les getters pour `code`, `solde` et `propriétaire`, mais pas de setters pour respecter la consigne.

```
public int getCode() { return code; }
public int getSolde() { return solde; }
public Client getProprietaire() { return proprietaire; }
```

8.

J’ai créé un constructeur qui initialise le code (auto-incrémenté), le solde à 0, et associe un propriétaire.

```
public Compte(Client proprietaire) {
    dernierCode++;
    this.code = dernierCode;
    this.solde = 0;
    this.proprietaire = proprietaire;
    System.out.println("Compte n°" + this.code + " créé pour " +
        proprietaire.getPrenom() + " " + proprietaire.getNom());
}
```

9.

Je n’est pas compris ce qui est demandé, je n'ai pas réussi à le faire

10.

Je n’est pas pu le faire car j’ai pas compris la question 9

Conclusion

Ce TP m’a vraiment aidé à mieux comprendre comment fonctionnent les classes et les méthodes en Java. J’ai eu quelques galères, surtout avec la question 9, mais au final, ça m’a permis de progresser et de mieux comprendre les concepts de la programmation orientée objet.